[Eric Lindemann]

# Music Synthesis with Reconstructive Phrase Modeling

[Capturing the dynamic behavior and expressivity of musical instruments]

Music synthesizers can be divided into two broad categories: functional synthesizers that generate sound from a mathematical formula or its electrical equivalent, and sampling synthesizers that generate sound by playing back recordings of musical sounds. The Theremin and the Ondes-Martenot were early examples of functional synthesizers dating from the 1920s. Electronic organs like the Hammond followed in the 1930s. Specialized electronic music studios were built in the 1950s and 1960s, followed by more compact analog synthesizers—the Moog, Buchla, Arp, Oberheim in the 1970s. In the late 1970s, digital functional synthesizers using FM technology became standard equipment for the working musician [1]. These functional synthesizers can be played expressively in live performance with physical controllers mapped directly to the parameters of the underlying mathematical formulas. However, the underlying sound generation formulae for these synthesizers are too simple to capture the complexity and richness of natural sounds.

In the late 1980s and 1990s physical modeling synthesizers were introduced that simulate the physics of natural instruments [2]. These functional synthesizers are capable of capturing the complex dynamics of natural instruments. However, physical modeling synthesizers can require a virtuoso mastery comparable to the original natural instrument. In order to use these models convincingly from a keyboard or directly from a composer's score requires a "physical model" of the performer as well as the instrument. The performer model has, as yet, not been incorporated in these systems.

The origins of sampling synthesis can be traced to the development of Musique Concrete in 1948 that used manual splicing of magnetic tape to create musical compositions [3]. A playable

1053-5888/07/$25.00©2007IEEE

keyboard instrument using magnetic tape—the Melotron—was developed in the 1960s. However, it wasn't until the 1980s that the ability to store sounds directly in computer RAM brought sampling into the mainstream. Today, samplers are the dominant synthesizer technology and an entire sample library industry has emerged. Samplers are often used to simulate orchestral instruments and competing orchestral sample libraries boast recordings of Stradivarius violins, famous orchestras performing in famous concert halls, and the highest bit rate digital recording techniques. With increasing compute power and storage capacity sample libraries have grown enormously in size from hundreds of kilobytes in the early 1980s to hundreds of gigabytes today.

The dominant sampler performance interface is the musical keyboard. The performer presses a key on the keyboard and a digital recording associated with that key is played back. There is no connection between the playback of one recording and the next and there is generally little the performer can do to the sound after a key is pressed. This approach can work well with percussion instruments but is not well suited to simulating expressive instruments such as violin or trumpet. The individual recordings can be stunningly realistic with all the complexity of natural sounds, but because of the lack of real-time performer interaction and the inability to model natural sounding transitions between notes, the results for expressive instruments often sound like a concatenation of separate sounds rather than a continuous musical phrase.

This article describes a new synthesis technology called reconstructive phrase modeling (RPM). A goal of RPM is to combine the realistic sound quality of sampling with the performance interaction of functional synthesis. Great importance is placed on capturing the dynamics of note transitions-slurs, legato, bow changes, etc. Expressive results are achieved with conventional keyboard controllers. Mastery of special performance techniques is not needed. RPM is used in a commercial software synthesizer called Synful Orchestra [4] that is used by composers and performers as a "virtual orchestra in a box." Synful Orchestra is the first of a planned series of products that will include jazz instruments, rock/funk instruments, ethnic instruments, and exploration of new sounds.

RPM is an analysis-synthesis system that is related to two important trends in computer music research. The first is a form of additive synthesis in which sounds are represented as a sum of time-varying harmonics plus noise elements. RPM creates expressive performances by searching a database of idiomatic instrumental phrases and combining modified fragments of these phrases to form a new expressive performance. This approach is related to another research trend called concatenative synthesis.

## ADDITIVE SYNTHESIS

The understanding that musical or pitched sounds can be represented as a series of pure-tone harmonics with integer multiple frequencies has its origins with Pythagoras and the Greeks. More recently we know that the mammalian ear and brain actively process sound in frequency bands [5], [6]. Modeling sound as a harmonic series maps well to both the physics of vibrating bodies like musical instruments and to the frequency-band processing of the human ear and brain. This helps to explain why a functional synthesizer like the Hammond organ whose control parameters are organized in frequency bands provides an intuitive interface for the performer.

In 1807 Jean-Baptiste Fourier discovered that *any* periodic function can be represented perfectly as a weighted sum of harmonically related sines and cosines [7]. (LaGrange, who served on Fourier's thesis committee, admired everything about Fourier's work *except* for the part about sines and cosines.) So, sinusoidal additive synthesis parameters are not only intuitive synthesizer controls but they also represent a hi-fidelity *coding strategy* for musical sound. It is natural to use this dual aspect of additive synthesis to bridge the gap between functional synthesizers, concerned with providing intuitive control parameters, and sampling synthesizers, concerned with recording and storing sound fragments.

If we assume a human audible bandwidth from 20 Hz to 20 kHz, then a low-pitched musical tone, say A at 110 Hz , will have a maximum of $20{,}000/110 = 181$ harmonics. Generally, low-pitched, naturally occurring tones will have a high-frequency roll-off, reducing the number of audible harmonics. Nevertheless, a general assumption of approximately 100 harmonics per tone is reasonable. Digital computers make it possible to generate these harmonics and to control their time-varying frequencies and amplitudes accurately. However, it is still computationally demanding. Several efforts to use inverse fast Fourier transform (IFFT) techniques to ease this computational burden have been described [8]–[13]. Other time-domain approaches to easing the computation burden have been "group additive synthesis" [14] and "multiple wavetable synthesis" [15].

The ability to digitally generate 100 harmonics per tone is a good thing. However, a composer presented with a computer tool allowing her to draw the time-varying amplitudes of 100 harmonics for each note of her composition would find it very difficult and extremely time consuming to design amplitude envelopes that resulted in a rich natural sound.

One approach to identifying the harmonic amplitude envelopes that lead to natural sound is to extract these envelopes from recordings of natural instruments. Analysis-synthesis systems are systems that transform a recorded sound to a set of time-varying parameters such as harmonic amplitudes and frequencies. Analysis-synthesis systems have a long history in telecommunications. For music, the first examples of analysis-synthesis based on additive parameters are from Risset and Mathews at Bell Labs in 1969 [16].

The fast Fourier transform (FFT) is an efficient recursive method for computing Fourier series coefficients. Applying the FFT to a recording of a single pitch period of a musical tone, say 401 samples of an A 110 tone sampled at 44,100 kHz, gives the complex amplitudes (or equivalently real amplitudes and phases) of the harmonics of the fundamental. However, this "pitch synchronous" approach requires accurate identification of pitch and alignment of FFT buffers. A more common approach first pioneered for synthesis by Jont Allen at Bell Labs [17] applies the FFT to overlapping "windowed" buffers of fixed length. This approach, referred to as the short-time Fourier transform (STFT), is equivalent to a filter bank with fixed linearly spaced overlapping frequency bands, where each band is sampled at a reduced "decimated" sample rate [18]. In this case a single pure tone sinusoid will appear in multiple bands due to the frequency band overlap. A sinusoid with time-varying frequency will move around between various fixed STFT filter-bank bands. Identifying a sinusoid in the STFT filter bank is further complicated by aliasing components that are generated due to decimation of the frequency bands. The problem of extracting harmonics or nonstationary sinusoids from the STFT has been addressed by [19]–[22].

Many musical sounds have noise components as well as pitched components. Examples include sustained bow noise, bow scratch during attacks, flute chiff at note attack, and sustained breath noise of a flute. While performing normal FFT-IFFT analysis-synthesis will preserve these noise elements, the extraction and resynthesis of harmonic amplitudes from the STFT filter bank will not. Several systems have addressed the problem of separation and resynthesis of noise elements using STFT techniques [23], [24].

RPM technology performs signal processing operations on each time-varying harmonic envelope as if it was a signal in its own right. For example, rapidly varying components of each envelope are separated from slowly varying components. A related technique based on wavelet analysis of harmonic envelopes is described in [25].

## CONCATENATIVE SYNTHESIS

RPM relies on a database of recorded musical phrases. Fragments from these phrases are taken from the database and spliced together or concatenated to form the musical output. The phrase-oriented approach contrasts with the traditional sample library approach of recording a collection of individual notes and mapping these to keys on the keyboard. The emphasis is on the connection between notes and capturing the many ways performers move from one note to the next. Notes do not stand alone. The timbre and contour of a note is directly affected by its phrase context. In speech synthesis this is referred to as "co-articulation." The pronunciation of a syllable depends on the syllables that precede and follow it. In this sense, RPM captures the co-articulation of musical sounds.

Concatenative speech synthesis (CSS) [26] is intended to capture co-articulation effects for human speech. With CSS a database of recorded speech fragments is maintained. These can be single speech phonemes, combinations of multiple sounds such as diphones or triphones, whole words, or even short phrases. Originally the emphasis was on highly structured databases with uniform collections of similar length sounds, but heterogeneous nonuniform databases with a variety of lengths and structures and corresponding unit selection and search strategies have also been the subject of research [27], [28]. Recordings in the database are concatenated to form complete phrases. There have also been a number of attempts to exploit concatenative synthesis for musical sounds [29]–[34]. Of particular note is the work on spectral modeling synthesis [35] incorporated in the Yamaha Vocaloid singing synthesis product.

RPM exploits a kind of MIDI control sequence-based database search and additive concatenative synthesis for rapidly varying components of the audio output signal. This concatenative stream is then combined with the slowly varying components, which are functionally generated directly from the MIDI control stream.

## OVERVIEW OF RPM

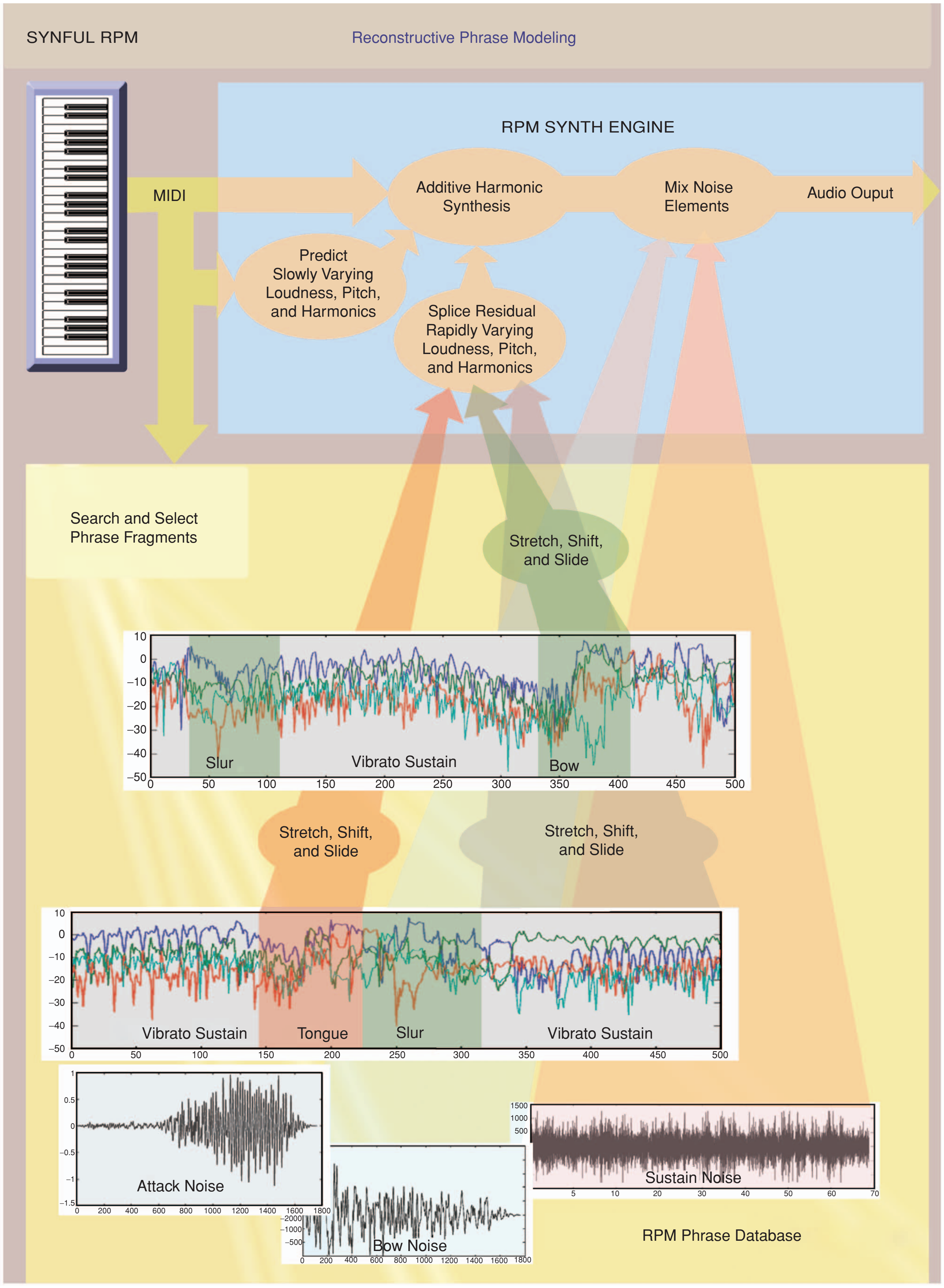
Figure 1 shows an overview of the RPM synthesizer. The input MIDI control stream may be generated automatically from a composer's score or may originate from a controller in real-time (keyboard, wind instrument, etc.). The stream includes note-on/note-off messages with associated pitch and velocity (an indication of intensity of the note) as well as continuous controller information that determines vibrato intensity, overall instrument loudness, timbre, and pitch-bend.

Slowly varying pitch and loudness controls are derived directly from the input MIDI control stream. MIDI volume/expression and velocity contribute to the loudness control, and pitch is derived from the MIDI note-on pitch and pitch-wheel control. In natural musical instruments the timbre of an instrument is highly correlated with pitch and loudness; e.g., when the performer blows harder the trumpet gets louder and its timbre gets brighter. RPM uses this correlation to predict a basic underlying timbre of the instrument based on the slowly varying pitch and loudness derived from the MIDI stream. This timbre is represented as slowly varying amplitudes of the individual harmonics of the instrument sound.

The slowly varying pitch and loudness and basic underlying timbre do not have the small rapid fluctuations and fine detail of the original instrument recordings. These rapid fluctuations in loudness, pitch, and timbre are what give the instrument its richness and realism. This is especially true during the transitions from one note to the next where the fluctuations are particularly rapid and idiosyncratic for each instrument.

To generate these rapid fluctuations RPM relies on a database of recorded musical phrases. These are not recordings of isolated notes as is typically the case with sample libraries but complete idiomatic musical passages that represent all kinds of articulation and phrasings: detached, slurred, portamento, sharp attacks, soft attacks, etc. The phrases are recorded in solo recording sessions one instrument at a time. For homogeneity, only one musician is used per instrument: one violinist, one horn player, etc.

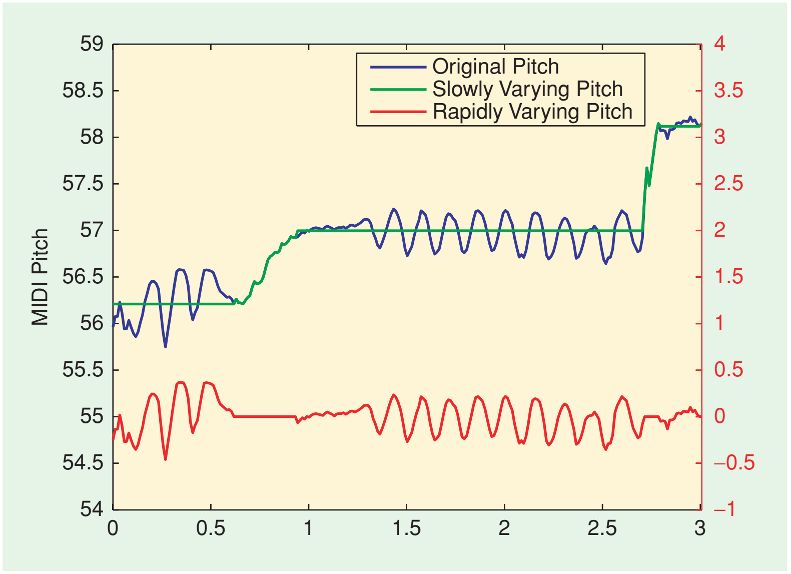
For Synful Orchestra, a recording session has something of the character of an orchestral audition. The musician is asked to play a variety of idiomatic passages from the orchestral literature. Taken together, these passages are intended to represent all the various ways the instrument is played. For orchestral playing there is a strong prior expectation for the sound and style of playing of each instrument. In future RPM-based products we will


SYNFUL RPM
Reconstructive Phrase Modeling
RPM SYNTH ENGINE
MIDI
Additive Harmonic Synthesis
Mix Noise Elements
Audio Ouput
Predict Slowly Varying Loudness, Pitch, and Harmonics
Splice Residual Rapidly Varying Loudness, Pitch, and Harmonics
Search and Select Phrase Fragments
Stretch, Shift, and Slide
Slur
Vibrato Sustain
Bow
Stretch, Shift, and Slide
Stretch, Shift, and Slide
Vibrato Sustain
Tongue
Slur
Vibrato Sustain
Attack Noise
Bow Noise
Sustain Noise
RPM Phrase Database

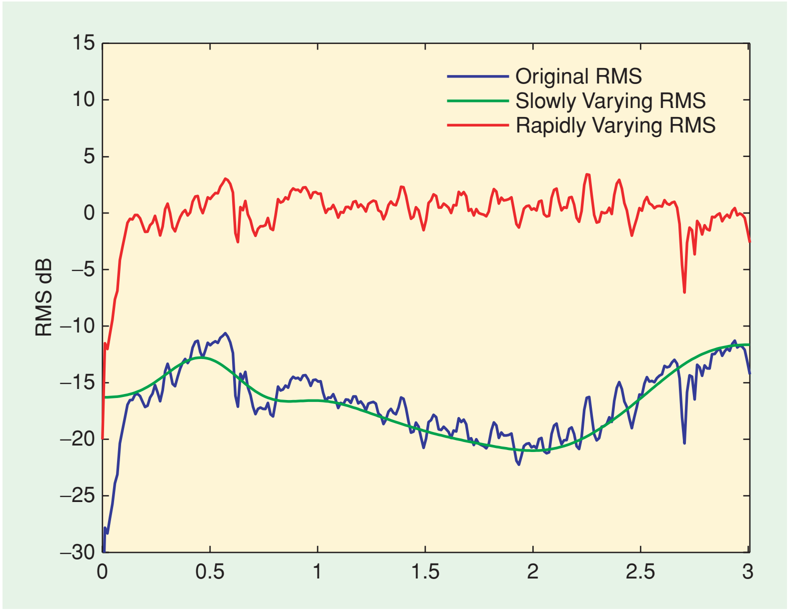
**[FIG1] Overview of the RPM synthesizer.**

emphasize jazz and popular instruments where the individual style of the performer can be more important. The basic sound and some of the aspects of phrasing of the individual performer are obviously part of the sound of the synthesizer. However, much of what we consider to be personal style is based on higher level phrasing control—the length of notes, the intensity of attack used in a particular musical context, and flexibility or rigidity in rhythmic interpretation. All of these higher level aspects of phrasing are under the control of the user of the RPM synthesizer, not the musician who performed the original phrases.

There is no attempt to systematize the contents of the RPM database; that is, to include performances of the same note or phrase fragment transposed to all pitches, or to include slurs from all pitches at all intervals in an organized pattern. We believe that the space of expressive articulations is much too large to be approached in this systematic manner and that any attempt to reduce the space to an organized performance matrix results in a significant reduction in expressive power. This reductionism is one of the fundamental problems with current sampling technology. Our approach is to capture a wide range of expressive playing using a "grab bag" approach and then to provide technological solutions that search the heterogeneous database and locate phrase fragments that fit the current musical context. Eliminating perceptual discontinuities while splicing these disparate phrase fragments is a significant challenge. We discuss solutions to this below.



**[FIG2] Original, slowly varying and rapidly varying pitch.**



**[FIG3] Original, slowly varying, and rapidly varying amplitude.**

The recorded phrases are stored in the database using a "residual pitch, loudness, and harmonics + noise" (RPLHN) representation. The RPLHN representation describes only the rapid fluctuations of the time-varying pitch, loudness, and harmonic amplitudes of the recorded sound. In addition, the RPLHN representation also separates noise elements such as breath noise and bow noise from the sound and stores these as pulse code modulation (PCM) sampled signals in the database. Figure 1 shows these different components in the database.

The phrases in the database are labeled with descriptor information that identifies the pitches, length of notes, intensity of notes, and type of note transitions—slurred, tongued, bowed, etc. Labeling of phrases is accomplished manually using a graphical editing tool.

Just as a skilled reader processes written text as phrases rather than individual words or syllables, a skilled instrumentalist processes groups of notes as musical phrases. The musical phrase forms a single shape or acoustic gesture in the mind of the performer. This shape is translated, almost unconsciously, into detailed physical actions on the instrument. Connecting notes to form phrases is essential to expressive performance.

In RPM the control stream is input to the "search and select phrases" (SSP) module of the synthesizer. This module mimics the role of the performer in forming phrases from the input control stream and in creating the bridge between the score/controller level detail of the input MIDI stream and the much more detailed acoustic image required for sound production.

The SSP module parses the input MIDI stream to locate musical phrase boundaries. When a phrase, consisting of a sequence on the order of two to eight notes, is identified, the SSP performs a search to find a "matching" phrase in the database. A match is determined by comparing information derived from the input MIDI sequence—pitch, note-duration, type of note transition, etc.—to information in the database phrase descriptors. Note transition type is determined from the input control sequence by a weighted combination of note velocity and the amount of time overlap between notes: a small overlap between the end of a note and the beginning of the subsequent note together with a small velocity indicates a legato or slurred note transition; a larger velocity or a small separation between notes indicates a detached (bowed or tongued) note transition.

Of course, it is unlikely that an exact match will be found in the database, so this search uses fuzzy matching criteria. Since there is not an exact match, the database phrase must be adjusted to correspond to the desired phrase. The operations include stretching or compressing the notes in time, shifting notes in pitch and intensity, sliding notes in time, modifying intensity and speed of vibrato, and adjusting the timbre of the recorded database phrase. These operations are carried out by the stretch, shift, and slide (SSS) operators shown in Figure 1. The flexible RPLHN representation makes these "morphing" operations relatively easy.

The morphed phrase from the database consists only of rapid fluctuations of pitch, loudness, and timbre. These are then superimposed on the slowly varying pitch, loudness, and timbre derived directly from the MIDI stream. This forms a complete representation of the time-varying loudness, pitch, and relative harmonic amplitudes of the musical phrase. This is then passed to the harmonic synthesis (HS) module where it is converted to a time-domain PCM output. In a parallel processing stream, noise elements are fetched from the database. These are already in PCM format. They are mixed with the output of the HS module to form the final audio output of the RPM synthesizer.

The term "reconstructive phrase modeling" is inspired from reconstructive surgery, where bits and pieces are taken from around the body (the database), molded, and spliced together (morphed) to rebuild a nose or mouth (the phrase).

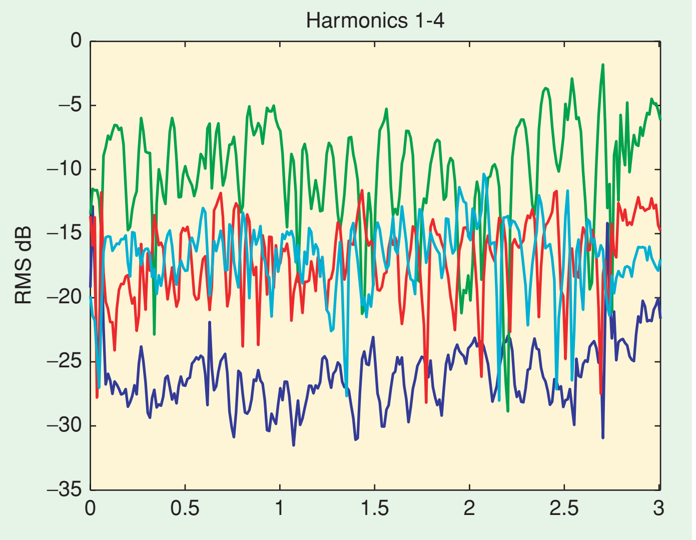
## RAPIDLY VARYING PITCH AND AMPLITUDE

RPM has an "at the factory" off-line analysis phase and a real-time synthesis phase. It is during analysis that the phrase database is constructed from original instrumental recordings and where statistical analyses are performed to provide parameters for timbre prediction. The synthesis phase occurs in real-time under direct control of the performer/composer. The analysis process calculates time-varying pitch, RMS amplitude, and relative harmonic amplitudes (harmonic amplitudes normalized by RMS amplitude) and separates noise components in a manner similar to the additive techniques described above. This is fully described in [36]–[38].

RPM combines slowly varying loudness, pitch, and harmonics components derived or predicted from the MIDI control stream with rapidly varying loudness, pitch, and harmonics components taken from the phrase database. The blue trace in Figure 2 shows the original time-varying pitch for a sequence of three violin notes that make up part of one of the database phrases. The red trace shows the rapidly varying pitch, which is derived during analysis from the original pitch by a note-oriented trend removal. The rapidly varying pitch is obtained by subtracting the mean pitch value of the nontransition part of each note together with a line that fits the transition part of the original pitch envelope from the original pitch envelope. The rapidly varying pitch is stored in the database.

During synthesis a slowly varying pitch is generated from the MIDI stream. The slowly varying pitch is a stair-step function of constant values reflecting the pitches of MIDI note-on messages. The steps are connected by curves derived from the MIDI pitch-wheel control. If the MIDI pitch wheel is not used, then the pitch curve connecting two notes is approximated by a line that fits the pitch transition in the original phrase. Often this pitch transition is an abrupt step from one pitch to another. The green trace in Figure 2 shows an example of slowly varying pitch. Typically, the slowly varying pitch derived from MIDI will have different note-durations than the rapidly varying pitch fetched from the database. So, the rapidly varying pitch is morphed to fit the slowly varying pitch. The compression or expansion of note durations is discussed below. The morphed rapidly varying pitch is added to the slowly varying pitch to form a complete time-varying pitch with complexity similar to the original phrase.

Figure 3 shows similar blue, green, and red traces for loudness. We use RMS amplitude in dB to represent loudness. True loudness is a more complicated function of the audio signal. However, for relatively homogeneous solo instrument recordings with restricted dynamic range, RMS amplitude is closely related to loudness. During synthesis, in a manner similar to the synthesis of time-varying pitch discussed above, a slowly varying RMS amplitude, as shown in green, is derived from MIDI velocity and continuous MIDI expression/volume control. This synthesized slowly varying RMS amplitude is added to the rapidly varying RMS amplitude fetched from the database (shown in red) to form the complete time-varying RMS amplitude used for synthesis.



**[FIG4] The first four original harmonics.**

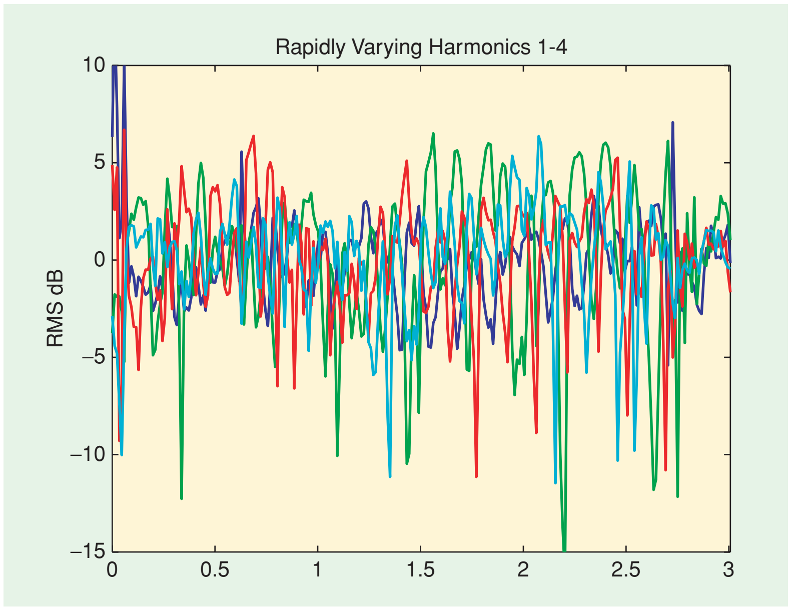
## SPECTRAL PREDICTION

Figure 4 shows time-varying amplitudes of the first four harmonics for the phrase of Figure 2. Figure 5 shows the corresponding rapidly varying harmonic components that are stored in the database. During synthesis the slowly varying pitch and RMS amplitude derived from the MIDI stream are used to predict a set of slowly varying harmonic amplitudes. The ability to predict harmonic amplitudes based on pitch and amplitude assumes a correlation between pitch and RMS amplitude and relative harmonic strength. The blue scatter dots in Figure 6 show the relationship between RMS amplitude and relative harmonic amplitude for the first harmonic of the French horn based on a number of recorded phrases. The first harmonic declines in relative strength with increasing RMS amplitude. Figure 7 shows the relationship between pitch and relative harmonic amplitude for the first harmonic. The relative strength of the first harmonic increases with pitch. The blue scatter dots in Figure 8 and Figure 9 show the same relationships for the fourth harmonic. As can be seen, the relative strength of the fourth harmonic has a more complex dependency on RMS amplitude but decreases in strength with respect to pitch. In Figure 7 and Figure 9 pitch is normalized over the useful range of the instrument, which in this case is the violin.

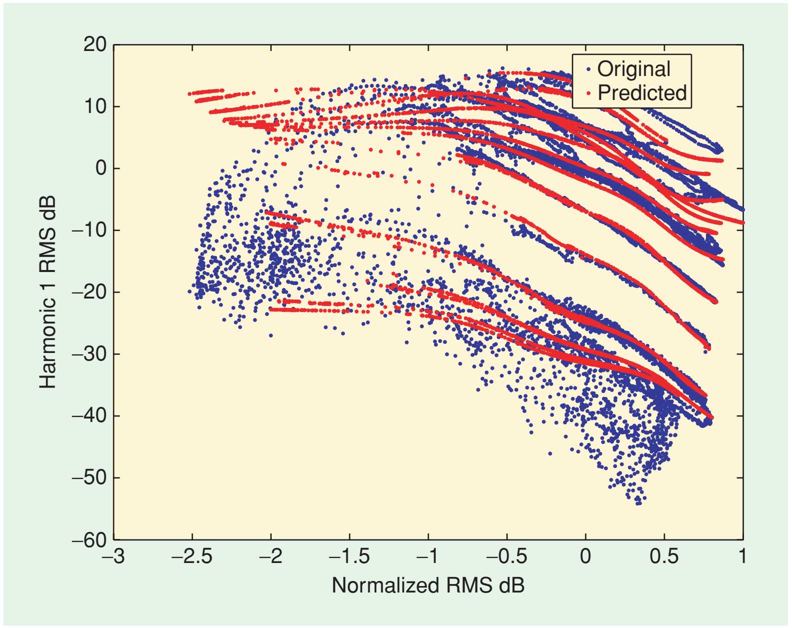
**A GOAL OF RPM IS TO COMBINE THE REALISTIC SOUND QUALITY OF SAMPLING WITH THE PERFORMANCE INTERACTION OF FUNCTIONAL SYNTHESIS.**

For each harmonic, a two-layer neural network with three hidden neurons [39] is used to predict the RMS amplitude for that harmonic given a pitch and RMS amplitude value pair. The neural network serves as an implementation of a nonlinear regression of harmonic amplitude on pitch and RMS amplitude [40]. The neural network is trained separately for each harmonic during analysis. The set of training inputs is the complete set of time-varying pitch and RMS amplitudes and the set of training outputs are relative harmonic amplitudes across all phrases for a particular instrument. The red dots in Figure 6 through Figure 9 show the predicted harmonic amplitudes from the neural network using smoothed versions of the pitch and RMS amplitude derived directly from the recorded phrases given as input.

Figure 10 shows the slowly varying components of the first four harmonics generated using the neural nets from the slowly varying pitch and RMS amplitude of Figures 2 and 3. During analysis. slowly varying harmonic amplitudes generated using the neural nets are subtracted from the overall time-varying harmonic amplitudes of Figure 4 to generate the rapidly varying harmonic amplitudes of Figure 5 that are stored in the phrase database. We refer to these rapidly varying harmonic amplitudes as harmonic residuals and the database signal representation as RPLHN. Owing to its highly structured sound representation, RPLHN is considerably more compact than the traditional PCM sampled representation. The Synful Orchestra library implemented using RPM is 32 MB in size and competes with sample libraries requiring many gigabytes.



[FIG5] Rapidly varying harmonic residuals.



[FIG6] First harmonic amplitude correlation.

## RAPID AND SLOW VARIATIONS

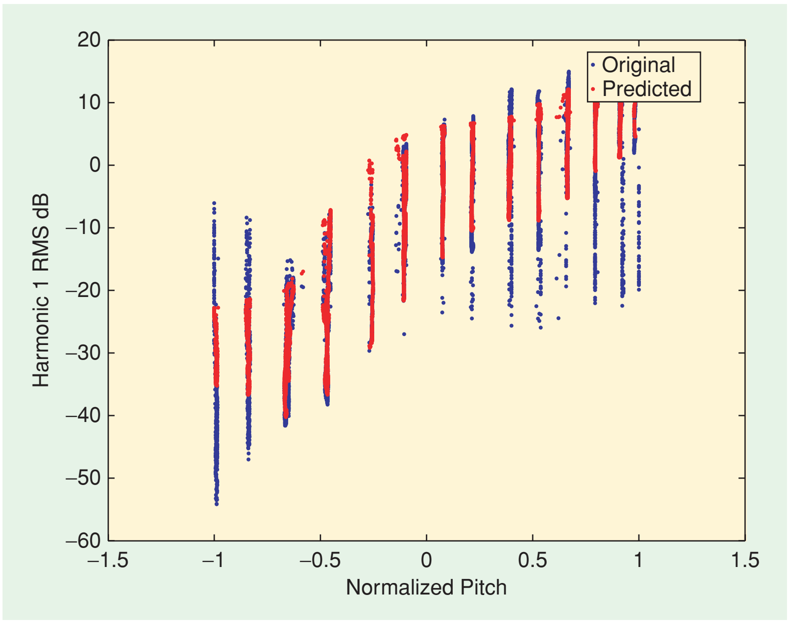
There are several advantages to separating slowly varying components from rapidly varying components. Since the slowly varying components are predicted directly from the MIDI control stream, they are immediately responsive to the performer. The generation of the slowly varying harmonic components is purely functional: given a pitch and loudness value pair, a harmonic series is generated. When the performer presses the volume or expression pedal, the underlying basic timbre of the instrument will immediately change in a manner similar to the change in timbre that occurs when blowing harder or altering bow speed/pressure. With this responsiveness the RPM synthesizer behaves with the immediacy of a functional synthesizer while the addition of rapidly varying components from the database preserves the realism and complexity of the original recordings.

The slowly varying components are responsible for the large majority of energy in the output audio signal. Since the rapidly varying components from the database represent a correspondingly smaller part of the signal energy, the morphing and splicing modifications applied to these rapidly varying components are less likely to produce noticeable distortion or perceived discontinuities.

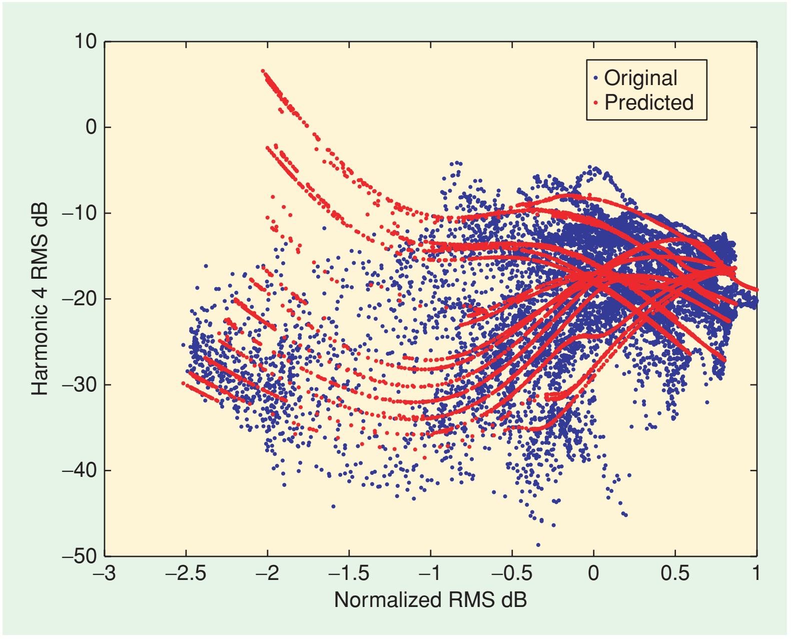
Traditional sample libraries often have individual recorded samples for every pitch. This is because changing the pitch of these recorded samples creates noticeable distortions in timbre, the so called "chipmunk effect." The rapidly varying harmonics in the database describe only the fluctuations of timbre around a basic underlying timbre generated from slowly varying harmonics. These rapidly varying components can be used over a relatively wide range of pitch and intensity since they do not affect the basic underlying timbre. This makes it possible to eliminate considerable redundancy from the database.

The rapidly varying harmonic components generally include variations related to vibrato. Vibrato is an oscillation in pitch and amplitude at about 6 Hz. The database section of Figure 1 shows several sections labeled "vibrato sustain." These segments show the time-varying harmonic amplitudes of the first four harmonics of the recorded phrase. Figure 4 shows the first four harmonics in greater detail. It is clear that in a natural vibrato all harmonic amplitudes do not oscillate in simple synchrony. The pattern across harmonics is quite complex with some harmonics oscillating at the perceived vibrato rate, other harmonics oscillating at twice the rate, and still others oscillating somewhat chaotically.

With traditional samplers, there are two ways of dealing with vibrato. In one method, the sample recording includes natural vibrato from the original performer and has all of the complexity of the harmonics shown in Figure 4. However the vibrato is fixed in the sample recording and cannot be controlled by the performer. In another method the recordings include no natural vibrato. Instead the vibrato is generated artificially by applying amplitude and frequency modulation during synthesis. However, this vibrato sounds artificial because it forces all harmonics to oscillate in simple synchrony. Both approaches are deficient.



[FIG7] First harmonic pitch correlation.



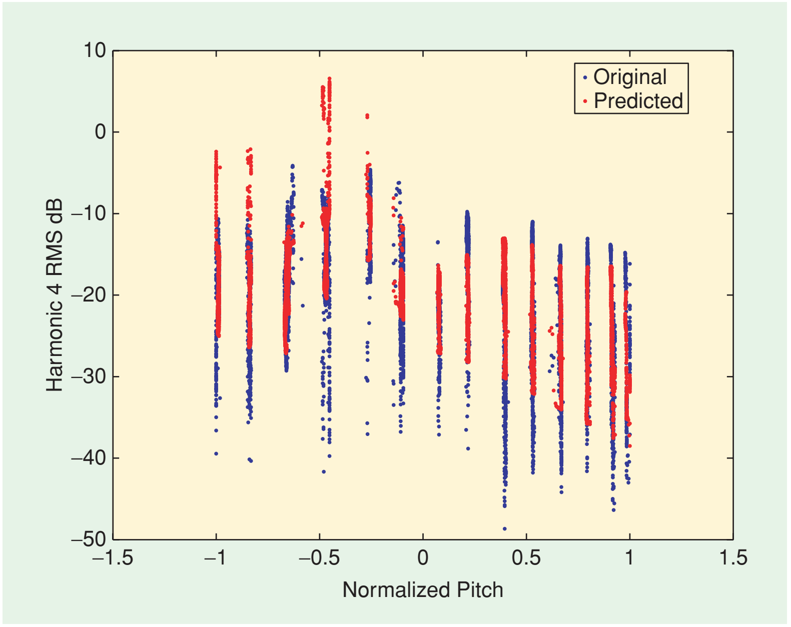
[FIG8] Fourth harmonic amplitude correlation.

With the RPM approach it is possible to provide real-time performance scaling of the rapidly varying vibrato components. The complex natural vibrato is preserved and controllable. Control over vibrato speed is also achievable by varying the readout rate of the rapidly varying components.

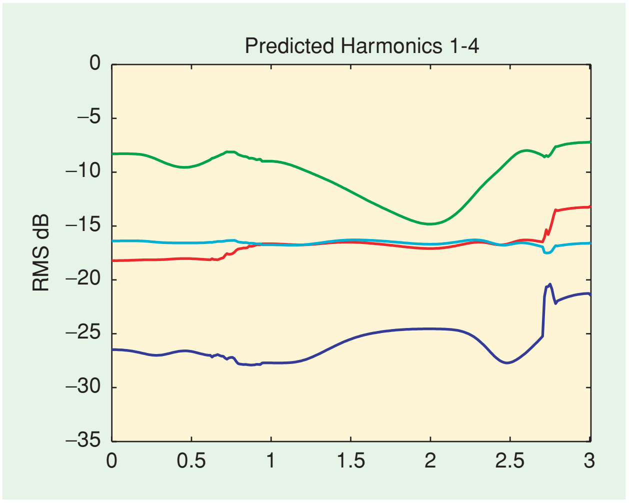
**ANALYSIS-SYNTHESIS SYSTEMS ARE SYSTEMS THAT TRANSFORM A RECORDED SOUND TO A SET OF TIME-VARYING PARAMETERS SUCH AS HARMONIC AMPLITUDES AND FREQUENCIES.**

## MULTIRESOLUTION CONTROLS

A skilled keyboard player trying to simulate a clarinet using RPM will use the volume/expression pedal to provide live performance gestures that mimic the breath control of the clarinetist (making small crescendos during certain note attacks, diminuendos at the end of long notes, etc.). However, the keyboard player will not vibrate her foot on the expression pedal at 6 Hz to simulate vibrato. This is a level of detail that is not included in the keyboard control stream. If the RPM synthesizer is controlled from an electronic woodwind controller, however, the MIDI control stream may very well contain 6-Hz oscillations associated with vibrato. At the other extreme, when receiving output directly from a composer's score (e.g., from a notation editor), the MIDI control stream will not include the kinds of breath-oriented gestures described above. The control stream will be considerably less expressive. Depending on the use case, the control stream provides different levels of expressive detail.



[FIG9] Fourth harmonic pitch correlation.



[FIG10] First four predicted harmonics.

The rapidly varying pitch and RMS amplitude are detrended versions of the time-varying pitch and RMS amplitude of the original recorded phrases. This detrending is in fact a form of high-pass filtering. To include vibrato in the rapidly varying components, the high-pass filter has low-frequency cutoff less than 6 Hz. However, as we continue to reduce this cutoff frequency to 3 Hz, 1 Hz, and 0.5 Hz, we include more of the original performance gestures in the rapidly varying components. This allows us to adjust the database as a function of the level of detail we anticipate receiving from the MIDI control stream. In fact, the database can be "multiresolution," including information from several high-pass cutoff frequencies, and thus supporting MIDI control streams with varying degrees of control detail corresponding to different use cases.

It is interesting to note that this approach does not require a detailed model of expressive control behavior. We assume that the recorded phrases in the database contain the necessary gestures and we adjust the level of control detail as a function of the detail we anticipate receiving from the MIDI control stream. This distinguishes RPM from attempts to enhance expressivity using performance rule systems that add human-like gestures to the control stream itself [41]–[44].

The separation of slowly varying and rapidly varying components and the prediction of slowly varying components directly from the control stream is described in detail in [37] and [38]. This prediction provides the responsiveness of a functional synthesizer. At the same time, the database of rapidly varying components preserves the richness and complexity of natural sound. Since the prediction of slowly varying components is based on intuitive pitch and loudness controls derived from the MIDI stream, the number of direct mapped control parameters is small and easily understood by the performer.

## PHRASE MATCHING, MORPHING, AND SPLICING

In order to parse the input MIDI stream, find phrase boundaries, and search the database for approximate matches, RPM needs to be able to look into the future of the MIDI stream to identify a note sequence before it is time for the notes to be played. When playing live from a controller, RPM has no advance knowledge of when a new note is coming, and so it does its best to react as expressively as possible with low latency when a new note occurs, using only past history of the input stream as a guide to phrasing—this is the RPM "live mode." However, once a performance is recorded in a sequencer, then on playback the future of the sequence is known and RPM can look-ahead to perform more complete phrase matching—this is the RPM "look-ahead" mode.

To identify the boundary between phrases in the input MIDI stream, RPM first looks for periods of silence. In the case of a series of short notes separated by short silence, RPM will group these together in a single phrase. As a phrase becomes longer, the ability to find close matches in the database is reduced. Therefore, the length of phrase searches is limited; e.g., a phrase search may include the first part of a long note but leave the next part of the note to the next phrase group. The phrase boundary defines the input target phrase that is used as a basis for database searching.

RPM forms a vector of parameters to describe the target phrase that includes note durations, overlap or separation between notes to characterize note transitions, pitches, velocities, and expression or volume pedal activity. The structure of this parameter vector is similar to the structure of the descriptors associated with phrases in the database. RPM first searches the database for phrases or pieces of phrases with the same number of notes as the target phrase and then forms the weighted sum of squares of differences between the target phrase parameters and the candidate database descriptor phrase parameters. This serves as a distance metric that is minimized to find the best phrase match in the database. The weighting is empirical. This kind of minimization of sum of squares metric is equivalent to a linear partition and category match in a very high dimensional vector space—in this case "phrase space" [45].

The note transition characteristics of the selected phrase (slurs, attacks, etc.) are assumed to match those of the target phrase since parameters are associated with these transitions are given heavy weighting in the distance metric. Since the pitch of the target phrase is set directly from the input MIDI stream through the generation of slowly varying pitch there is no pitch adjustment that needs to be applied to the rapidly varying pitch fetched from the database, which describes only fluctuations around the slowly varying pitch. The same considerations apply to rapidly varying RMS amplitude and relative harmonic amplitudes fetched from the database.

The note-durations of the selected phrase generally do not match those of the target phrase. Note lengths are shortened or lengthened by deleting or repeating (in a quasi-random manner) segments of the rapidly varying components. In the case of notes with vibrato, random segment repetition is performed on vibrato period boundaries. These vibrato period boundaries are labeled during analysis. The operations of deletion and repetition as well as the concatenation of one phrase to the next require splicing pitch, RMS amplitude, and harmonic sequences. This splicing must not introduce noticeable discontinuities in audio output. This is especially true since splices often take place in the middle of notes where discontinuities are particularly noticeable.

[ THE PHRASE-ORIENTED APPROACH CONTRASTS WITH THE TRADITIONAL SAMPLE LIBRARY APPROACH OF RECORDING A COLLECTION OF INDIVIDUAL NOTES AND MAPPING THESE TO KEYS ON THE KEYBOARD. ]

If the database consisted of PCM sampled sounds there would be no way to avoid discontinuities. Even if well-constructed cross-fade splices were used there would still be noticeable phase cancellation and changes of timbre across the splice. Conventional additive synthesis removes the problem of phase cancellation but still results in abrupt changes of timbre when phrases from different parts of the database are spliced. With RPM only rapidly varying components are spliced. The slowly varying underlying pitch, RMS amplitude, and harmonics are completely continuous across the splice.

For long notes, on the order of 1 s or more, we select splice points at about three-quarters of the way through the note, and we splice to an appropriate length transition (e.g., ¼ second) leading into the following note. For shorter notes we choose splice points closer to the middle of a note so that we capture the full transition between notes where dynamic behavior is most evident. Whenever possible, splices are avoided by selecting multinote sequences from the database. The emphasis is not on the optimal selection of splice points but on the design of a system that can tolerate splices from any note to any other note with similar characteristics. The separation of rapidly varying and slowly varying functionally generated components is important in making this splicing process robust.

As discussed above, searching and selecting elements from the database is based on matching criteria derived from the input MIDI stream (pitch, intensity, note duration, note overlap, etc.). These represent an extremely simple description of musical sound characteristics. These are the same characteristics that are used in the descriptors associated with notes and phrases in the database. It is tempting to include a wider range of descriptive information in the database descriptors: perceived acoustic roughness, brightness, more complete descriptions of slur characteristics such as portamento, etc. However, these elements are not readily derived from the standard MIDI input. Our aim is to provide good instrumental interpretations from limited control information. This is similar to the relationship between conductor/composer and musician. The composer provides a score with limited descriptive information and the performer is expected to provide an expressive interpretation in keeping with the established tradition for their instrument. From a practical standpoint, we have found with Synful Orchestra that it is often difficult to convince users to include even standard continuous controls such as volume or expression in their MIDI sequences. This experience reinforces our bias toward simple control interfaces that deliver maximum musical results. RPM database searching, morphing, and splicing are discussed in greater in detail in [37] and [38].

**[ WE BELIEVE THAT THE SPACE OF EXPRESSIVE ARTICULATIONS IS MUCH TOO LARGE TO BE APPROACHED IN A SYSTEMATIC MANNER AND THAT ANY ATTEMPT TO REDUCE THE SPACE TO AN ORGANIZED PERFORMANCE MATRIX RESULTS IN A SIGNIFICANT REDUCTION IN EXPRESSIVE POWER. ]**

## EFFICIENT TIME-DOMAIN ADDITIVE SYNTHESIS

Humans have a reduced ability to resolve individual time-varying harmonic amplitudes at high-frequencies. This is part of the psycho-acoustic theory of critical bands [5], [6]. RPM exploits this by using a vector quantized wave-table synthesis at high frequencies while using a more conventional sinusoidal oscillator bank at low frequencies.

As described above, transient and sustain noises are separated from the audio recordings using harmonic analysis. The remaining audio signal consists of a sum of sinusoidal harmonics. We call this the tonal audio signal. During harmonic analysis, the time-varying frequencies, amplitudes, and phases of the sinusoidal harmonic components of the tonal audio signal are determined. The 20 lowest frequency time-varying harmonics form the low-frequency harmonic sequence (LFHS) that encodes the low-frequency part of the tonal audio signal.

The remaining high-frequency time-varying harmonics form the high-frequency harmonic sequence (HFHS). We then proceed to vector quantize (VQ) the HFHS. This means that we find a small set (e.g., $<150$) of high-frequency harmonic vectors that can be used to represent the many thousands of high-frequency harmonic vectors that would otherwise be in the database. The small number of high-frequency harmonic vectors form a HFHVQ codebook. The idea is that for any high-frequency harmonic vector in the original HFHS there is a vector in the HFHVQ codebook that is not too far away (using a sum of squares distance measure) from the vector in the HFHS. That vector in the HFHS is then encoded as an index (HFHVQ_IX) into the HFHVQ codebook. To form this HFHVQ codebook a generalized Lloyd VQ algorithm is used, as described in [46].

Each vector in the HFHVQ codebook is inverse Fourier transformed (IFT) to form a single pitch period time-domain oscillator table with a high-frequency-only spectrum corresponding to the harmonic vector in the HFHVQ codebook. A random phase response is assigned to the HFHVQ vector before the IFT. The same random phase response is assigned to all vectors in the HFHVQ codebook. This means that the corresponding time-domain oscillator tables can be cross faded without causing phase cancellation. The collection of inverse transformed time-domain oscillator tables forms a high-frequency time-domain vector quantization (HFTDVQ) codebook. It is the HFTDVQ codebook that is stored along with the database.

For high-frequency synthesis above the 20th harmonic, a single table lookup oscillator using the table from the HFTDVQ codebook is used to generate all high harmonics. The table used generally changes from synthesis frame to synthesis frame (about every 10 ms). The table selected is determined by the HFHVQ_IX stored in the database for that synthesis frame. The ability to use one table lookup oscillator for approximately 80 high-frequency harmonics represents a considerable reduction in computation compared with 80 separate sinusoidal oscillators. The low-frequency 20 harmonics are synthesized using separate time-domain table-lookup oscillators. The details of the VQ process and corresponding synthesis are described in detail in [36].

## MIXING NOISE ELEMENTS

During analysis RPM separates noise elements from the original recorded phrases. There are two types of noise elements: transient noises occurring during attacks and note transitions, and sustain noises occurring during note sustains. These noise components are stored as traditional PCM samples. During analysis, RPM determines the level of the noises in the original recordings relative to the pitched or harmonic part of the sound. During synthesis, noise elements are fetched from the database and mixed with the newly synthesized harmonic components at relative levels determined during analysis. Other than gain there are no transformations applied to the noise elements. Additional performance control over noise levels is provided to allow for "scratchier" sounding strings, sharper attacks, etc. Sustain noises are played with traditional sampler cross-fade looping applied when time extension is required.

## CONCLUSION

RPM achieves musical expressivity through a blend of functional additive synthesis and phrase-oriented parametric concatenative synthesis. The realism and complexity of natural sound is preserved by maintaining a database of rapidly varying components derived from original recordings. By including complete phrases, the database emphasizes transitions between notes and phrasing gestures that span several notes.

Slowly varying components are derived directly from the MIDI control stream so that dynamic changes in timbre occur in response to standard MIDI performance controls. By varying the frequency boundary between rapidly and slowly varying components, RPM remains expressive in live performance and when rendering directly from a composer's score. Many Synful Orchestra RPM sound examples can be heard by visiting www.synful.com. In future developments, RPM will leverage its flexible RPLHN sound representation and spectral prediction methodology to explore the creation of new "fictional" instruments though hybridization and mutation of existing RPM instrument databases.

## AUTHOR

*Eric Lindemann* (eric@synful.com), founder of Synful, has a long background in music and technology. He led the design of the IRCAM Signal Processing Workstation that was used in computer music facilities around the world. He designed DSP microprocessors for Cirrus Logic and participated in the design of the first fully programmable DSP hearing aid for GN Resound. He was a founding engineer of Waveframe, makers of high-end digital audio workstations. He studied music composition with Nadia Boulanger, Olivier Messiaen, and Iannis Xenakis; has played keyboard for numerous movie scores; and has toured with pop groups including the Fifth Dimension and Osmond Brothers.

## REFERENCES

[1] J. Chowning, "The synthesis of complex audio spectra by means of frequency modulation," *J. Audio Eng. Soc.*, vol. 21, no. 7, 1973.

[2] J.O. Smith, "Music applications of digital waveguides," Center for Computer Research in Music and Acoustics (CCRMA), Stanford Univ., CCRMA Tech. Report STAN-M-39, 1987.

[3] P. Schaeffer, *Traite des Objets Musicaux*, 1st ed. Paris, France: Editions du Seuil, 1966.

[4] E. Lindemann, *Synful Orchestra*, www.synful.com, 2004–2006.

[5] H. Fletcher, *Speech and Hearing in Communication*, J, Allen, Ed., Acoustical Society of America, 1995.

[6] B.C.J. Moore, *An Introduction to the Psychology of Hearing*. New York: Academic, 1989.

[7] S. Hawking, *God Created the Integers*. Running Press, 2005. p. 496.

[8] A. Freed, "Inverse transform narrow band/broad band sound synthesis," U.S. Patent 5686683, 1997.

[9] A. Dembo and D. Malah, "Signal synthesis from modified discrete short-time transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-36, pp. 168–181, Feb. 1988.

[10] J. Laroche, "Synthesis of sinusoids via non-overlapping inverse Fourier transform," *IEEE Trans. Speech and Audio Processing*, vol. 8, pp. 471–477, July 2000.

[11] X. Rodet and P. Depalle, "Spectral envelopes and inverse FFT synthesis," in *Proc. 93rd Conv. Audio Engineering Society*, San Francisco, CA, Preprint 3393 (H-3), 1992.

[12] M. Goodwin and X. Rodet, "Efficient Fourier synthesis of nonstationary sinusoids," presented at *Int. Computer Music Conf.*, 1994.

[13] R.J. McAulay and T.F. Quatieri, "Computationally efficient sine-wave synthesis and its application to sinusoidal transform coding," presented at *IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, New York, NY, 1998.

[14] P. Kleczkowski, "Group additive synthesis," *Computer Music J.*, vol. 13, no. 1, pp. c12–20, 1989.

[15] A. Horner, J. Beauchamp, and L. Haken, "Methods for multiple wavetable synthesis of musical instrument tones," *J. Audio Eng. Soc.*, vol. 41, no. 5, May 1993.

[16] J.-C. Risset and M. Mathews, "Analysis of instrument tones," *Physics Today*, vol. 22, no. 2, pp. 23–40, 1969.

[17] J.B. Allen, "Short term spectral analysis, synthesis, and modification by discrete Fourier transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, no. 3, pp. 235–238, 1977.

[18] P.P. Vaidyanathan, *Multirate Systems and Filter Banks*. New York: Prentice-Hall 1993.

[19] F.J. Harris, "On the use of windows for harmonic analysis with the discrete Fourier transform," *Proc. IEEE*, vol. 66, pp. 51–83, 1978.

[20] J.O. Smith and X. Serra, "PARSHL: An analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation," in *Proc. Int. Computer Music Conf., Tokyo, Japan, 1987*.

[21] R.J. McAulay and T.F. Quatieri, "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Transactions Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 744–754, Aug. 1986.

[22] T.F. Quatieri and R.J. McAulay, "Audio signal processing based on sinusoidal analysis/synthesis," in *Applications of DSP to Audio & Acoustics*, Boston: Kluwer, pp. 343–416, 1998.

[23] X. Serra and J.O. Smith, "Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition," *Computer Music J.*, vol. 14, pp. 12–24, 1990.

[24] J. Laroche, Y. Stylianou, and E. Moulines, "HNS: Speech modification based on a harmonic + noise model," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Minneapolis, MN, 1993.

[25] G. Evangelista, "Pitch-synchronous wavelet representations of speech and music signals," *IEEE Trans. Signal Processing*, vol. 41, no. 12, Dec. 1993.

[26] T. Dutoit, *An Introduction to Text-to-Speech Synthesis*. Norwell, MA: Kluwer, 1997.

[27] A.J. Hunt, A.W. Black, "Unit selection in a concatenative speech synthesis system using a large speech database," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Atlanta, GA, pp. 373–376, May 1996.

[28] Y. Sagisaka, "Speech synthesis by rule using an optimal selection of non-uniform synthesis units," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, New York, NY, pp. 679–682, 1988.

[29] R. Kobayashi, "Sound clustering synthesis using spectral data," in *Proc. Int. Computer Music Conf.*, Singapore, 2003.

[30] G. Beller, Gregory, D. Schwarz, T. Hueber, and X. Rodet, "A hybrid concatenative synthesis system on the intersection of music and speech," in *Journees d'Informatique Musicale (JIM)*, St. Denis, France: MSH Paris Nord, 2005.

[31] J. Bonada and X. Serra, "Synthesis of the singing voice by performance sampling and spectral models," *IEEE Signal Processing Mag.*, vol. 24, no. 2, 67–79, Mar. 2007.

[32] D. Schwarz, "Corpus-based concatenative synthesis," *IEEE Signal Processing Mag.*, vol. 24, no. 2, 92–104, Mar. 2007.

[33] D. Schwarz, "A system for data-driven concatenative sound synthesis," in *Proc. COST-G6 Conf. Digital Audio Effects (DAFx)*, Verona, Italy, pp. 97–102, Dec. 2, 2000.

[34] M. Puckette, "Low-dimensional parameter mapping using spectral envelopes," in *Proc. Int. Computer Music Conf.*, Miami, Florida. Nov. 2004.

[35] J. Bonada, O. Celma, A. Loscos, J. Ortol, and X. Serra, "Singing voice synthesis combining excitation plus resonance and sinusoidal plus residual models," in *Proc. ICMC*, Havana, Cuba, 2001.

[36] E. Lindemann, "Encoding and synthesis of tonal audio signals using dominant sinusoids and a vector quantized residual tonal signal," U.S. Patent 6298322, May 6, 1999.

[37] E. Lindemann, "Music synthesizer capable of expressive phrasing," U.S. Patent 6316710, Sept. 27, 1999.

[38] E. Lindemann, "Audio signal synthesis system based on probabilistic estimation of time-varying spectra," U.S. Patent 6111183, Sept. 7, 1999.

[39] S. Haykin, *Neural Networks, A Comprehensive Foundation*. New York: Macmillan, 1994.

[40] V.N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer, 2000.

[41] M. Clynes, "SuperConductor—The Global Music Interpretation and Performance Program." Available: www.superconductor.com.

[42] R. Bresin and A. Friberg, "Emotional coloring of computer-controlled music performances," *Computer Music J.*, vol. 24, no. 4, pp. 44–63, 2000.

[43] A. Friberg, "pDM: An expressive sequencer with real-time control of the KTH music performance rules," *Computer Music J.*, vol. 30, no. 1, pp. 37–48, 2006.

[44] M. Laurson, V. Norilo, and M. Kuuskankare, "PWGLSynth: A visual synthesis language for virtual instrument design and control," *Computer Music J.*, vol. 29, no. 3, pp. 29–41, 2005.

[45] N. Cristianini and J. Shawe-Taylor, *Vector Support Machines*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[46] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA: Kluwer, 1992.